Order-Reducing Conjugate Gradients versus Block AOR for Constrained Least-Squares Problems

Douglas James*
Department of Mathematics
North Carolina State University
Raleigh, North Carolina 27695-8205

Dedicated to Gene Golub, Richard Varga, and David Young

Submitted by Michael Neumann

ABSTRACT

We compare the convergence properties of two iterative algorithms for solving equality-constrained least-squares problems. The first algorithm, due to Barlow, Nichols, and Plemmons, applies a variation of the conjugate-gradient algorithm to a symmetric positive definite system which is smaller than the original problem. The second, block accelerated overrelaxation, is a two-parameter generalization of block SOR. Barlow, Nichols, and Plemmons have proven that their order-reducing conjugate-gradient algorithm converges faster than block SOR. We extend their result to show that the algorithm is also superior to block AOR. Numerical experiments on some structural engineering problems support the analysis.

1. INTRODUCTION

Our interest is the solution of the *equality-constrained least-squares problem*, or *problem LSE*: given an $m_1 \times n$ matrix E, an $m_2 \times n$ matrix G, an $m_1 \times 1$ vector b, and an $m_2 \times 1$ vector c,

$$\text{minimize} \quad \|Gy - c\|_2 \qquad \text{such that} \quad Ey = b. \tag{1}$$

We will assume that E has full row rank, so b is in the range of E and the problem has at least one solution. We will also require that the nullspaces

*Graduate studies sponsored by the Dept. of Mathematical Sciences, US Air Force Academy, Colo., and funded by the Air Force Institute of Technology, WPAFB, Ohio. Research directed by Robert J. Plemmons under Air Force grant AFOSR-88-0285.

655 Avenue of the Americas, New York, NY 10010 0024-3795/91/$3.50

of E and G intersect trivially (or, equivalently, that $\begin{bmatrix} E \\ G \end{bmatrix}$ has full column rank), guaranteeing that the solution is unique. Finally, we presume that the matrices are large and sparse, so that it is plausible to consider iterative algorithms.

A number of essentially equivalent formulations of LSE will prove helpful:

CONSTRAINED MINIMIZATION PROBLEM. Let $F = G^T G$ and $s = -G^T c$. Note that $\|Gy - c\|_2^2 = y^T F y + 2y^T s + c^T c$, and that the term $c^T c$ has no effect on the value of y at which the quantity is minimized. Hence, LSE is equivalent to the problem

$$\text{minimize} \quad y^T F y + 2y^T s \qquad \text{such that} \quad Ey = b. \tag{2}$$

KUHN-TUCKER FORMULATION. Introduce the Lagrange multiplier λ and the residual $r = c - Gy$. Then solving LSE amounts to finding y, r, and λ satisfying

$$\begin{bmatrix} E & 0 & 0 \\ G & I & 0 \\ 0 & G^T & E^T \end{bmatrix} \begin{bmatrix} y \\ r \\ \lambda \end{bmatrix} = \begin{bmatrix} b \\ c \\ 0 \end{bmatrix}. \tag{3}$$

SADDLE-POINT FORMULATION. Introduce the Lagrange multiplier μ into the constrained minimization formulation. Then one can find y satisfying (2) by solving

$$\begin{bmatrix} -F & E^T \\ E & 0 \end{bmatrix} \begin{bmatrix} y \\ \mu \end{bmatrix} = \begin{bmatrix} s \\ b \end{bmatrix}. \tag{4}$$

The multiplier μ is related to λ in the Kuhn-Tucker equations by $\mu = -\lambda$.

The motivating example is the static analysis of engineering structures: given a large structure subjected to an external load, find the internal forces at equilibrium. When the problem is modeled using the *force method* (see, for example, [8]), the constraint $Ey = b$ captures the fact that the forces sum to zero at any node in the structure (the *equilibrium condition*). The vector y represents the unknown internal forces, b is the vector of external loads, and

E (commonly called the *equilibrium matrix*) is determined by the shape and connectivity of the structure. The vector $-\lambda$ represents the nodal displacements. We seek the particular solution to the constraint which minimizes the *complementary energy*; the matrix F, which helps define the energy functional, is determined by the material properties of the elements in the structure. In a rigid bar truss, F is a diagonal matrix with positive diagonal entries determined by Hooke's law (see Strang [18]). In more general structures, F is block-diagonal, with one small block for every element in the structure. The blocks in F are always symmetric nonnegative definite, and are symmetric positive definite when all elements behave elastically (see, for example, Przemieniecki [16]). Unless the elements are prestressed, the vectors c and s are zero in this application.

Static analysis of engineering structures is but one example of a more general physical principle at work. Minimizing an energy functional subject to an equilibrium constraint is a central idea throughout the physical sciences (see Strang [17, 18]), so problem LSE in its various forms has a wide range of applications. In most cases, the mathematical models associated with these applications satisfy the assumptions under which we proceed.

Both of the algorithms we consider begin with a modified form of the Kuhn-Tucker equations, in which the diagonal blocks are square, nonsingular, and easily invertible (see Section 2 below). The first algorithm, which we call *Algorithm BNP*, was initially proposed and analyzed by Barlow, Nichols, and Plemmons [1]. The authors rewrite the modified Kuhn-Tucker system by applying block Gauss elimination, isolating a symmetric positive definite subproblem in the process. They then apply a variant of the conjugate-gradient algorithm to this subproblem, generating at each iteration an approximation to the original unknown y. The algorithm is *dimension-* or *order-reducing* in the sense that the subproblem involves fewer unknowns than problem LSE itself. The second algorithm, known as *block accelerated overrelaxation* or *AOR* [7], is a two-parameter generalization of block SOR.

Barlow, Nichols, and Plemmons [1], extending work by Freund [6], have shown that BNP applied to problem LSE is superior to block SOR in a certain well-defined sense. However, Papadopoulou, Saridakis, and Papatheodorou [13] argue that under certain technical conditions, a three-block version of AOR may outperform the most important version of block SOR (see Section 3). Our goal here is to extend the work in [1] and [6], proving that BNP is superior to block AOR as well.

Section 2 introduces algorithm BNP as it is derived in [1], and summarizes some of its properties. In Section 3, we overview the SOR and AOR algorithms. The main result is the subject of the next two sections: in Section 4, we prove that BNP applied to ordinary least-squares problems is superior to block AOR, then extend the result to constrained problems in Section 5. In

Section 6 we provide some numerical experiments comparing the algorithms on structural engineering problems, and offer some concluding remarks.

2. ALGORITHM BNP

It is difficult to apply traditional iterative methods to the Kuhn-Tucker equations as written in (3): the diagonal blocks are not even square, let alone nonsingular. We therefore start by repartitioning these equations. Since we are assuming E has full row rank, and that $\begin{bmatrix} E \\ G \end{bmatrix}$ has full column rank, we can reorder the rows of G and c, and repartition

$$G = \begin{bmatrix} G_1 \\ G_2 \end{bmatrix}$$

so that

$$A_1 = \begin{bmatrix} E \\ G_1 \end{bmatrix} \tag{5}$$

is square and nonsingular. Defining $A_2 = G_2$ for convenience, we now have

$$\begin{bmatrix} E \\ G \end{bmatrix} = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix}. \tag{6}$$

Make these substitutions in (3) and reorder the column blocks to obtain the *modified Kuhn-Tucker equations*:

$$\begin{bmatrix} A_1 & 0 & K \\ A_2 & I & 0 \\ 0 & A_2^T & A_1^T \end{bmatrix} \begin{bmatrix} y \\ r_2 \\ z_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ c_2 \\ 0 \end{bmatrix}, \tag{7}$$

where

$$c = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}, \qquad r = \begin{bmatrix} r_1 \\ r_2 \end{bmatrix}, \qquad b_1 = \begin{bmatrix} b \\ c_1 \end{bmatrix}, \qquad z_3 = \begin{bmatrix} \lambda \\ r_1 \end{bmatrix},$$

and

$$K = \begin{bmatrix} 0 & 0 \\ 0 & I \end{bmatrix}.$$

Note that the diagonal blocks are now square and nonsingular.

We address elsewhere (see [11]) the problem of selecting the augmentation matrix G_1 to produce a convenient A_1. For our purposes it is enough to know that when G has full column rank (as it does in elastic analysis and many other applications), it is relatively easy to modify the problem to produce an upper triangular A_1. The task is somewhat more difficult when G lacks full column rank [2]; see [9] for a class of algorithms suitable for such problems.

The derivation of BNP in [1] begins with this repartitioned system. The authors first apply block elimination to reduce the system to block upper triangular form. The lowest of four blocks in the resulting system is the symmetric positive definite system

$$(I + Y^T Y) r_1 = h, \qquad \text{where} \quad Y = A_2 A_1^{-1} \bar{K}, \quad \bar{K} = \begin{bmatrix} 0 \\ I \end{bmatrix}. \tag{8}$$

The right-hand-side vector is given by $h = Y^T(A_2 A_1^{-1} b_1 - c_2)$.

The unknown r_1 consists of the $n - m_1$ leading components of the residual $c - Gy$. In principle, one can solve this reduced system for r_1, then use back substitution on the transformed Kuhn-Tucker system to recover the remaining unknowns. In practice, this is not necessary: the analysis in [1] shows that a variant of conjugate gradients applied to the reduced system produces as by-products all that is needed to recover the full solution y. It is this order-reducing conjugate-gradient algorithm which we call BNP.

We summarize the complete algorithm below. This particular outline obscures opportunities to avoid redundant calculations, but will prove useful for the analysis in Section 4. See the original paper [1] for a version suitable for implementation.

ALGORITHM 1: BNP (problem LSE).

1. Initialize:
   (a) $y^{(0)}$ arbitrary (normally $y^{(0)} = A_1^{-1} b_1$).
   (b) $r_1^{(0)} = \bar{K}^T(b_1 - A_1 y^{(0)}) = c_1 - G_1 y^{(0)}$ (normally $r_1^{(0)} = 0$).
   (c) $r_2^{(0)} = c_2 - A_2 y^{(0)} = c_2 - G_2 y^{(0)}$.
   (d) $\nu_0 = r_1^{(0)} + Y^T r_2^{(0)}$ [ν is the defect $(I + Y^T Y) r_1 - h$].
   (e) $s_0 = \nu_0$ (s is the direction vector).

2. For $k = 0, 1, \ldots,$ until $\nu_k^T \nu_k <$ tolerance:
   (a) $\gamma_k = \nu_k^T \nu_k / s_k^T (I + Y^T Y) s_k$.
   (b) $r_1^{(k+1)} = r_1^{(k)} - \gamma_k s_k$.
   (c) $y^{(k+1)} = y^{(k)} + \gamma_k A_1^{-1} \overline{K} s_k$.
   (d) $r_2^{(k+1)} = r_2^{(k)} - \gamma_k Y s_k$.
   (e) $\nu_{k+1} = r_1^{(k+1)} + Y^T r_2^{(k+1)}$.
   (f) $\beta_{k+1} = \nu_{k+1}^T \nu_{k+1} / \nu_k^T \nu_k$.
   (g) $s_{k+1} = \nu_{k+1} + \beta_{k+1} s_k$.

It's worth noting that the symmetric positive definite matrix $I + Y^T Y$ is likely to have a number of desirable properties. In particular, the scalar $\lambda = 1$ is often a multiple eigenvalue due to column-rank deficiencies in Y. This, of course, limits the maximum number of conjugate-gradient iterations required for convergence.

3. p-CYCLIC SOR AND BLOCK AOR

Given a nonsingular linear system $Cz = f$, let $C = D - L - U$ define a splitting of the coefficient matrix into block diagonal, lower triangular, and upper triangular parts respectively. The well-known block SOR algorithm is then defined by an iteration involving a relaxation parameter ω:

$$(D - \omega L) z^{(k+1)} = [(1 - \omega) D + \omega U] z^{(k)} + \omega f. \tag{9}$$

If we consider the modified Kuhn-Tucker system (7), two plausible choices of the block-diagonal matrix D produce three- and two-block SOR methods respectively:

$$D_3 = \begin{bmatrix} A_1 & & \\ & I & \\ & & A_1^T \end{bmatrix} \quad \text{and} \quad D_2 = \begin{bmatrix} A_1 & & \\ A_2 & I & \\ & & A_1^T \end{bmatrix}. \tag{10}$$

While little is known about the optimal iteration parameter ω for arbitrary linear systems, the modified Kuhn-Tucker system enjoys some special properties which make a more complete analysis possible. When the coefficient matrix of the modified Kuhn-Tucker system is partitioned using either D_3 or D_2, it is a so-called *p-cyclic* matrix, and the associated SOR algorithms are known as *p-cyclic SOR methods*. An elegant theory, due

largely to Young [21] and Varga [19, 20], relates the spectrum of the SOR iteration matrix to that of the corresponding Jacobi iteration matrix. Exploiting this p-cyclic theory and special properties of the Jacobi iteration matrices allows one to obtain a number of important results for p-cyclic SOR on the modified Kuhn-Tucker equations. In particular, Plemmons [15] has established that 2-cyclic SOR applied to LSE converges for sufficiently small values of ω (there may or may not be values of ω for which 3-cyclic SOR converges). Additionally, Markham, Neumann, and Plemmons [12] have shown that the asymptotic convergence of optimal 2-cyclic SOR is superior to 3-cyclic SOR. Pierce, Hadjidimos, and Plemmons [14] and Eiermann, Niethammer, and Ruttan [5] establish results for more general problems, demonstrating the importance of the special properties of the modified Kuhn-Tucker system.

Despite the elegant convergence theory for p-cyclic SOR, BNP is superior in exact arithmetic. This was established by Freund [6] for unconstrained least-squares problems, and by Barlow, Nichols, and Plemmons [1] for equality-constrained problems. More precisely, if $y^{(0)}$, $r_2^{(0)} = c_2 - G_2 y^{(0)}$, and $r_1^{(0)} = c_1 - G_1 y^{(0)}$ serve as initial iterates for both BNP and 2-cyclic SOR, then the iterates at subsequent steps satisfy the inequality

$$\|c - Gy_{\mathrm{BNP}}^{(k)}\|_2 \leqslant \|c - Gy_{\mathrm{SOR}}^{(k+1)}\|_2. \tag{11}$$

We note that the work required to complete an iteration of each of these algorithms is essentially the same. This means that, at least in theory, BNP should prove faster than optimal 2-cyclic SOR on problem LSE.

Now define the two-parameter generalization of SOR known as *accelerated overrelaxation* or *AOR* (see, for example, Hadjidimos [7]):

$$(D - \beta L) z^{(k+1)} = [(1-\omega) D + (\omega - \beta) L + \omega U] z^{(k)} + \omega f. \tag{12}$$

Note that the special case $\omega = \beta$ is block SOR.

Again we consider the blockings given by (10), referring to the corresponding algorithms as 3-AOR and 2-AOR respectively. The optimal choice of parameters for 2-AOR occurs somewhere on the line $\omega = \beta$ (see Papadopoulou, Saridakis, and Papatheodorou [13]), so optimal 2-AOR coincides with optimal 2-cyclic SOR. Thus BNP is superior to 2-AOR in exact arithmetic. Optimal 3-AOR, however, does not necessarily occur when $\omega = \beta$. In fact, Papadopoulou et al. establish in [13] that under some very restrictive (and highly technical) conditions, the asymptotic convergence of 3-AOR may be better than optimal 2-AOR (and therefore 2-cyclic SOR). For

these reasons, we limit our attention in Section 4 to 3-AOR, and establish a result similar to Equation (11).

We will need the explicit description of 3-AOR outlined below. Remember that the matrices K and $\overline{K}$ are defined in Equations (7) and (8).

ALGORITHM 2: 3-AOR (problem LSE).

1. Initialize:
   (a) $y^{(0)}$ arbitrary.
   (b) $r_2^{(0)} = c_2 - A_2 y^{(0)}$.
   (c) $r_1^{(0)} = c_1 - G_1 y^{(0)}$.
   (d) $\lambda^{(0)}$ arbitrary.
   (e) $z_3^{(0)} = \begin{bmatrix} \lambda^{(0)} \\ r_1^{(0)} \end{bmatrix}$.
2. For $k = 0, 1, \ldots,$ until convergence:
   (a) $y^{(k+\frac{1}{2})} = A_1^{-1}(b_1 - Kz_3^{(k)}) = A_1^{-1}(b_1 - \overline{K}r_1^{(k)})$.
   (b) $r_2^{(k+\frac{1}{2})} = c_2 - A_2\left[(1-\beta)y^{(k)} + \beta y^{(k+\frac{1}{2})}\right]$.
   (c) $z_3^{(k+\frac{1}{2})} = -A_1^{-T}A_2^T\left[(1-\beta)r_2^{(k)} + \beta r_2^{(k+\frac{1}{2})}\right]$.
   (d) $y^{(k+1)} = (1-\omega)y^{(k)} + \omega y^{(k+\frac{1}{2})}$.
   (e) $r_1^{(k+1)} = (1-\omega)r_1^{(k)} + \omega r_1^{(k+\frac{1}{2})}$.
   (f) $z_3^{(k+1)} = (1-\omega)z_3^{(k)} + \omega z_3^{(k+\frac{1}{2})}$.

Technically, we could initialize $r_1^{(0)}$ and $r_2^{(0)}$ to arbitrary values. As defined above, however, the initial values are both mathematically plausible and consistent with the BNP iteration (unlike BNP, however, subsequent iterates do not satisfy $r_1^{(k)} = c_1 - G_1 y^{(k)}$ or $r_2^{(k)} = c_2 - A_2 y^{(k)}$). In any case, our experiments suggest the method is not particularly sensitive to the choice of initial iterates.

4. BNP VERSUS AOR: ORDINARY LEAST-SQUARES

Our goal is to establish that BNP applied to problem LSE is superior to 3-AOR in exact arithmetic, by proving a result analogous to Equation (11). The proof follows the spirit of the arguments in Freund [6] and Barlow, Nichols, and Plemmons [1], and proceeds in three steps:

1. Given an ordinary least-squares problem (no constraints) and an arbitrary starting vector for algorithm BNP, the iterates minimize the 2-norm of the residual vector in a sequence of Krylov spaces.

2. Given an ordinary least-squares problem and the same starting vector used in algorithm BNP, the AOR iterates lie in the same Krylov spaces as the BNP iterates, regardless of the choice of AOR parameters. Thus, the residual vectors associated with the BNP iterates are no larger in norm than those generated by 3-AOR.
3. Given a constrained least-squares problem, there is a related ordinary least-squares problem with the following property: when BNP is applied to both problems, the residual vectors at each iteration are equal. Similarly, 3-AOR applied to the constrained and unconstrained problems generates the same sequence of residual iterates. Hence, by step 2, the residual vectors associated with the BNP iterates are no larger in norm than those generated by 3-AOR.

We complete steps 1 and 2 in this section. In the next section, we relate the constrained problem to an ordinary least-squares problem to complete step 3. To prevent confusion, we use a tilde ($\tilde{}$) over many of the quantities associated with the unconstrained problem of this section.

Begin with the ordinary least-squares problem

$$\text{minimize} \quad \|\tilde{A}x - \tilde{b}\|_2, \tag{13}$$

where $\tilde{A}$ has full column rank. View this problem as a "constrained" least-squares problem with zero constraints. The choice of x as the unknown will prove convenient in the next section.

By analogy with (5), partition the coefficient matrix

$$\tilde{A} = \begin{bmatrix} \tilde{A}_1 \\ \tilde{A}_2 \end{bmatrix}$$

(reordering the rows of $\tilde{A}$ and $\tilde{b}$ as necessary) so that $\tilde{A}_1$ is square and nonsingular. Partition the vector

$$\tilde{b} = \begin{bmatrix} \tilde{b}_1 \\ \tilde{b}_2 \end{bmatrix}$$

compatibly. The resulting modified Kuhn-Tucker equations are similar to those in (7), except the identity matrix replaces the matrix K, and there is no

Lagrange multiplier:

$$\begin{bmatrix} \tilde{A}_1 & 0 & I \\ \tilde{A}_2 & I & 0 \\ 0 & \tilde{A}_2^T & \tilde{A}_1^T \end{bmatrix} \begin{bmatrix} x \\ \tilde{r}_2 \\ \tilde{r}_1 \end{bmatrix} = \begin{bmatrix} \tilde{b}_1 \\ \tilde{b}_2 \\ 0 \end{bmatrix}. \tag{14}$$

Note that BNP and 3-AOR as outlined in Sections 2 and 3 are perfectly well defined for this system. To obtain a version of BNP for ordinary least-squares problems, simply replace $\bar{K}$ with the identity matrix, and substitute $\tilde{A}_1$, $\tilde{A}_2$, $\tilde{b}_1$, $\tilde{b}_2$, $\tilde{r}_1$, $\tilde{r}_2$, and x for A_1, A_2, b_1, c_2, r_1, r_2, and y respectively. The AOR algorithm is just as simple to modify, but the absence of z_3 changes its appearance somewhat. For convenience, we describe it explicitly.

ALGORITHM 3: 3-AOR (ordinary least squares).

1. Initialize:
   (a) $x^{(0)}$ arbitrary.
   (b) $\tilde{r}_2^{(0)} = \tilde{b}_2 - \tilde{A}_2 x^{(0)}$.
   (c) $\tilde{r}_1^{(0)} = \tilde{b}_1 - \tilde{A}_1 x^{(0)}$.
2. For $k = 0, 1, \ldots,$ until convergence:
   (a) $x^{(k+\frac{1}{2})} = \tilde{A}_1^{-1}(\tilde{b}_1 - \tilde{r}_1^{(k)})$.
   (b) $\tilde{r}_2^{(k+\frac{1}{2})} = \tilde{b}_2 - \tilde{A}_2\left[(1-\beta)x^{(k)} + \beta x^{(k+\frac{1}{2})}\right]$.
   (c) $\tilde{r}_1^{(k+\frac{1}{2})} = -\tilde{A}_1^{-T}\tilde{A}_2^T\left[(1-\beta)\tilde{r}_2^{(k)} + \beta\tilde{r}_2^{(k+\frac{1}{2})}\right]$.
   (d) $x^{(k+1)} = (1-\omega)x^{(k)} + \omega x^{(k+\frac{1}{2})}$.
   (e) $\tilde{r}_2^{(k+1)} = (1-\omega)\tilde{r}_2^{(k)} + \omega\tilde{r}_2^{(k+\frac{1}{2})}$.
   (f) $\tilde{r}_1^{(k+1)} = (1-\omega)\tilde{r}_1^{(k)} + \omega\tilde{r}_1^{(k+\frac{1}{2})}$.

In the discussion below, we use the following notation: if v is a vector, and S a set of vectors, then $v + S$ represents the set $\{v + s : s \in S\}$. Similarly, if B is a matrix, then BS is the set $\{Bs : s \in S\}$. If S is convex, so are $v + S$ and BS; if S is a vector subspace, BS is a subspace as well.

Let the initial iterates for both BNP and 3-AOR be as in Algorithm 3. For convenience, define

$$\tilde{Y} = \tilde{A}_2\tilde{A}_1^{-1}. \tag{15}$$

Now define the following vectors in $\mathscr{R}^n$:

$$\hat{v}_0 = \tilde{b}_1 + \tilde{Y}^T \tilde{b}_2, \tag{16}$$

$$v_0 = \hat{v}_0 - (\tilde{A}_1 + \tilde{Y}^T \tilde{A}_2) x^{(0)}, \tag{17}$$

$$\begin{aligned} \hat{w}_0 &= \tilde{A}_1^{-1} \hat{v}_0 \\ &= \tilde{A}_1^{-1} (\tilde{b}_1 + \tilde{Y}^T \tilde{b}_2), \end{aligned} \tag{18}$$

$$\begin{aligned} w_0 &= \tilde{A}_1^{-1} v_0 \\ &= \hat{w}_0 - \tilde{A}_1^{-1} (\tilde{A}_1 + \tilde{Y}^T \tilde{A}_2) x^{(0)}. \end{aligned} \tag{19}$$

Additionally, let

$$\tilde{C} = \tilde{A}_1^{-1} \tilde{Y}^T \tilde{Y} \tilde{A}_1 = \tilde{A}_1^{-1} \tilde{Y}^T \tilde{A}_2, \tag{20}$$

noting that

$$w_0 = \hat{w}_0 - (I + \tilde{C}) x^{(0)}. \tag{21}$$

Finally, define a sequence of Krylov subspaces:

$$\begin{aligned} W_0 &= \{0\}, \\ W_k &= \operatorname{span}\{w_0, \tilde{C} w_0, \ldots, \tilde{C}^{k-1} w_0\}. \end{aligned} \tag{22}$$

LEMMA 1 (Freund). *Let* $x^{(0)}$, $\tilde{r}_2^{(0)} = \tilde{b}_2 - \tilde{A}_2 x^{(0)}$, *and* $\tilde{r}_1^{(0)} = \tilde{b}_1 - \tilde{A}_1 x^{(0)}$ *be initial iterates for algorithm BNP applied to the ordinary least-squares problem* (13). *Then the* kth *iterate* $x^{(k)}$ *lies in the set* $x^{(0)} + W_k$. *Moreover,* $x^{(k)}$ *minimizes the residual* $\|\tilde{r}\|_2 = \|\tilde{b} - \tilde{A}x\|_2$ *over all* x *in* $x^{(0)} + W_k$.

Proof. See [6]. ■

THEOREM 1. *Let* $x^{(0)}$, $\tilde{r}_2^{(0)} = \tilde{b}_2 - \tilde{A}_2 x^{(0)}$, *and* $\tilde{r}_1^{(0)} = \tilde{b}_1 - \tilde{A}_1 x^{(0)}$ *be initial iterates for both BNP and 3-AOR applied to the ordinary least squares problem* (13). *Let* $x_{\text{BNP}}^{(k)}$ *and* $x_{\text{AOR}}^{(k)}$ *represent the* kth *iterates generated by the*

two algorithms respectively, and define $\tilde{r}^{(k)}_{\text{BNP}} = \tilde{b} - \tilde{A}x^{(k)}_{\text{BNP}}$ *and* $\tilde{r}^{(k)}_{\text{AOR}} = \tilde{b} - \tilde{A}x^{(k)}_{\text{AOR}}$ *to be the associated residuals. Then the iterates satisfy the inequality*

$$\|\tilde{r}^{(k)}_{\text{BNP}}\|_2 \leqslant \|\tilde{r}^{(k+1)}_{\text{AOR}}\|_2$$

in exact arithmetic, regardless of the AOR iteration parameters.

Proof. By Lemma 1, it suffices to prove that the AOR iterate $x^{(k+1)}_{\text{AOR}}$ lies in $x^{(0)} + W_k$. Suppressing the "AOR" subscripts for simplicity, we accomplish this by establishing the following relationships:

(a) $x^{(k)} \in x^{(0)} + W_{k-1}$,
(b) $\tilde{b}_2 - \tilde{r}_2^{(k-\frac{1}{2})} \in \tilde{A}_2(x^{(0)} + W_{k-1})$,
(c) $\tilde{b}_2 - \tilde{r}_2^{(k)} \in \tilde{A}_2(x^{(0)} + W_{k-1})$,
(d) $\tilde{b}_1 + \tilde{Y}^T\tilde{r}_2^{(k-\frac{1}{2})} \in \hat{v}_0 - \tilde{Y}^T\tilde{A}_2(x^{(0)} + W_{k-1})$,
(e) $\tilde{b}_1 + \tilde{Y}^T\tilde{r}_2^{(k)} \in \hat{v}_0 - \tilde{Y}^T\tilde{A}_2(x^{(0)} + W_{k-1})$,
(f) $\tilde{b}_1 - \tilde{r}_1^{(k-\frac{1}{2})} \in \hat{v}_0 - \tilde{Y}^T\tilde{A}_2(x^{(0)} + W_{k-1})$,
(g) $\tilde{b}_1 - \tilde{r}_1^{(k)} \in \tilde{A}_1x^{(0)} + \text{span}\{v_0\} + \tilde{A}_2\tilde{Y}^T W_{k-1}$,
(h) $x^{(k+\frac{1}{2})} \in x^{(0)} + W_k$.

Our goal is to establish the first of these relationships; the others are means toward that end. We argue by induction. Listing the early iterates explicitly, we have:

$$x^{(\frac{1}{2})} = x^{(0)},$$

$$\tilde{r}_2^{(\frac{1}{2})} = \tilde{r}_2^{(0)},$$

$$\tilde{r}_1^{(\frac{1}{2})} = -\tilde{Y}^T\tilde{r}_2^{(0)};$$

$$x^{(1)} = x^{(0)},$$

$$\tilde{r}_2^{(1)} = \tilde{r}_2^{(0)},$$

$$\tilde{r}_1^{(1)} = \tilde{r}_1^{(0)} - \omega v_0;$$

$$x^{(1+\frac{1}{2})} = x^{(0)} + \omega w_0.$$

Use these values to confirm that each of the inductive hypotheses holds for

$k = 1$. Now assume all eight hypotheses hold for a fixed k, and consider $k+1$.

Proof of (a): Since $x^{(0)} + W_{k-1}$ is contained in $x^{(0)} + W_k$, inductive assumption (a) tells us that $x^{(k)}$ is an element of $x^{(0)} + W_k$. By assumption (h), $x^{(k+\frac{1}{2})}$ is also in this set. Thus, by convexity, $x^{(k+1)} = (1-\omega)x^{(k)} + \omega x^{(k+\frac{1}{2})}$ is in $x^{(0)} + W_k$ as required.

Proof of (b): Equation 2(b) of Algorithm 3 tells us that

$$\tilde{b}_2 - \tilde{r}_2^{(k+\frac{1}{2})} = \tilde{A}_2\left[(1-\beta)x^{(k)} + \beta x^{(k+\frac{1}{2})}\right].$$

By assumption (a), we know $\tilde{A}_2 x^{(k)}$ is contained in $\tilde{A}_2(x^{(0)} + W_{k-1})$, which in turn is a subset of $\tilde{A}_2(x^{(0)} + W_k)$. Moreover, $\tilde{A}_2 x^{(k+\frac{1}{2})}$ is in the latter set by inductive assumption (h). Thus, by convexity, $\tilde{b}_2 - \tilde{r}_2^{(k+\frac{1}{2})}$ is in $\tilde{A}_2(x^{(0)} + W_k)$ as required.

Proof of (c): From assumption (c) and the containment argument used above, we have that $\tilde{b}_2 - \tilde{r}_2^{(k)}$ is in $\tilde{A}_2(x^{(0)} + W_k)$. The proof of (b) tells us that $\tilde{b}_2 - \tilde{r}_2^{(k+\frac{1}{2})}$ is in the latter set as well. Rearrange Equation 2(e) of Algorithm 3 to see that $\tilde{b}_2 - \tilde{r}_2^{(k+1)}$ is a convex combination of $\tilde{b}_2 - \tilde{r}_2^{(k)}$ and $\tilde{b}_2 - \tilde{r}_2^{(k+\frac{1}{2})}$, and so is in $\tilde{A}_2(x^{(0)} + W_k)$ as required.

Proof of (d): From the definition of $\hat{v}_0$ in Equation (16), and the definition of $\tilde{r}_2^{(k+\frac{1}{2})}$ in Algorithm 3, we find that

$$\tilde{b}_1 + \tilde{Y}^T \tilde{r}_2^{(k+\frac{1}{2})} = \hat{v}_0 - \tilde{Y}^T \tilde{A}_2\left[(1-\beta)x^{(k)} + \beta x^{(k+\frac{1}{2})}\right].$$

Assumptions (a) and (h) combined with convexity then give us the required result.

Proof of (e) *and* (f): Similar to the proof of (c).

Proof of (g): By assumption (g), there is a scalar α, and a vector z_1 in $\tilde{Y}^T \tilde{A}_2 W_k$, such that

$$\tilde{b}_1 - \tilde{r}_1^{(k)} = \tilde{A}_1 x^{(0)} + \alpha v_0 + z_1.$$

By the proof of (f), there is a $z_2 \in \tilde{Y}^T \tilde{A}_2 W_k$ such that

$$\tilde{b}_1 - \tilde{r}_1^{(k+\frac{1}{2})} = \hat{v}_0 - \tilde{Y}^T \tilde{A}_2 x^{(0)} + z_2.$$

Let $z = (1-\omega)z_1 + \omega z_2$, and note that $z \in \tilde{Y}^T \tilde{A}_2 W_k$. Rearrange Equation 2(f) of Algorithm 3 to obtain

$$\tilde{b}_1 - \tilde{r}_1^{(k+1)} = (1-\omega)\left(\tilde{b}_1 - \tilde{r}_1^{(k)}\right) + \omega\left(\tilde{b}_1 - \tilde{r}_1^{(k+\frac{1}{2})}\right).$$

Remembering that $v_0 = \hat{v}_0 - (\tilde{A}_1 + \tilde{Y}^T\tilde{A}_2)x^{(0)}$, use the expressions for $\tilde{b}_1 - \tilde{r}_1^{(k)}$ and $\tilde{b}_1 - \tilde{r}_1^{(k+\frac{1}{2})}$ to find that

$$\tilde{b}_1 - \tilde{r}_1^{(k+1)} = \tilde{A}_1 x^{(0)} + \alpha(1-\omega)v_0 + \omega v_0 + z,$$

which is in $\tilde{A}_1 x^{(0)} + \text{span}\{v_0\} + \tilde{Y}^T\tilde{A}_2 W_k$ as required.

Proof of (h): From Algorithm 3 we have $x^{(k+1+\frac{1}{2})} = \tilde{A}_1^{-1}(\tilde{b}_1 - \tilde{r}_1^{(k+1)})$. By the proof of (g), there exists a scalar γ such that

$$\tilde{b}_1 - \tilde{r}_1^{(k+1)} \in \tilde{A}_1 x^{(0)} + \gamma v_0 + \tilde{Y}^T\tilde{A}_2 W_k.$$

Recalling that $w_0 = \tilde{A}_1^{-1}v_0$, these two facts tell us that $x^{(k+1+\frac{1}{2})}$ is in the set $x^{(0)} + \gamma w_0 + \tilde{A}_1^{-1}\tilde{Y}^T\tilde{A}_2 W_k$. But w_0 is an element of W_{k+1}, and $\tilde{A}_1^{-1}\tilde{Y}^T\tilde{A}_2 W_k = \tilde{C}W_k$, which is a vector subspace of W_{k+1}. Hence $x^{(k+1+\frac{1}{2})}$ is in $x^{(0)} + W_{k+1}$ as required. ■

5. BNP VERSUS AOR: CONSTRAINED LEAST SQUARES

In the previous section, we proved that BNP applied to an ordinary least-squares problem converges at least as fast as 3-AOR in exact arithmetic. To extend this result to constrained least-squares problems, we establish a connection between problem LSE and an ordinary least-squares problem. The argument below is an adaptation of Theorem 2.1 in Barlow, Nichols, and Plemmons [1], and is based on the classical nullspace method.

Consider problem LSE as given in Equation (1), and the associated modified Kuhn-Tucker equations (7). Define the matrix

$$N = A_1^{-1}\overline{K} = A_1^{-1}\begin{bmatrix} 0 \\ I \end{bmatrix}. \tag{23}$$

Observe that

$$\begin{bmatrix} E \\ G_1 \end{bmatrix} A_1^{-1} = A_1 A_1^{-1} = \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix}. \tag{24}$$

This means $EA_1^{-1} = [I\ 0]$, so $EN = 0$. Moreover, N has $n - m_1$ linearly independent columns, so the columns of N form a basis for the nullspace of E.

By a similar argument, the vector

$$y_p = A_1^{-1} b_1 \tag{25}$$

is a particular solution of the constraint $Ey = b$. Note that any vector satisfying the constraint can be written as $y = y_p + Nx$ for some choice of x, and minimizing $\|Gy - c\|_2$ subject to the constraint amounts to minimizing $\|G(y_p + Nx) - c\|_2$ over all possible x. The latter minimization is an unconstrained problem of order $n - m_1$. Isolating the unknown x, we have the ordinary least-squares problem

$$\text{minimize} \quad \|\tilde{A}x - \tilde{b}\|_2 \qquad \text{where} \quad \tilde{A} = GN, \quad \tilde{b} = c - Gy_p. \tag{26}$$

Referring to Equation (24), we find that

$$GN = \begin{bmatrix} G_1 N \\ G_2 N \end{bmatrix} = \begin{bmatrix} I \\ Y \end{bmatrix}, \tag{27}$$

where $Y = A_2 A_1^{-1} \bar{K}$ is as in the BNP system (8). Additionally,

$$G_1 y_p = G_1 A_1^{-1} b_1 = \bar{K}^T \begin{bmatrix} b \\ c_1 \end{bmatrix} = c_1, \tag{28}$$

so $c_1 - G_1 y_p = 0$. All of these relationships give us an elegant form for the modified Kuhn-Tucker system associated with the ordinary least-squares problem (26):

$$\begin{bmatrix} I & 0 & I \\ Y & 0 & 0 \\ 0 & Y^T & I \end{bmatrix} \begin{bmatrix} x \\ \tilde{r}_2 \\ \tilde{r}_1 \end{bmatrix} = \begin{bmatrix} 0 \\ c_2 - G_2 y_p \\ 0 \end{bmatrix}. \tag{29}$$

In terms of the notation from the previous section, we have

$$\tilde{A}_1 = I, \tag{30}$$

$$\tilde{A}_2 = Y, \tag{31}$$

$$\tilde{b}_1 = 0, \tag{32}$$

$$\tilde{b}_2 = c_2 - A_2 A_1^{-1} b_1. \tag{33}$$

Now let $y^{(0)}$ be an arbitrary initial iterate for both BNP and 3-AOR applied to problem LSE. There is a unique $x^{(0)}$ such that $y^{(0)} = y_p + Nx^{(0)}$; in fact, the correct value is given by

$$x^{(0)} = \overline{K}^T\left(A_1 y^{(0)} - b_1\right). \tag{34}$$

Use this value of $x^{(0)}$ as the initial iterate for BNP and 3-AOR applied to the ordinary least-squares problem (26). We now define the corresponding initial residual iterates, and relate the subsequent iterates for the constrained problem to those of the unconstrained problem.

LEMMA 2. *Let* $y^{(0)}$, $r_2^{(0)} = c_2 - A_2 y^{(0)}$, *and* $r_1^{(0)} = c_1 - G_1 y^{(0)}$ *be initial iterates for BNP applied to problem LSE. Define* $x^{(0)}$ *as in Equation* (34), *and let* $x^{(0)}$, $\tilde{r}_2^{(0)} = \tilde{b}_2 - \tilde{A}_2 x^{(0)}$, *and* $\tilde{r}_1^{(0)} = \tilde{b}_1 - \tilde{A}_1 x^{(0)}$ *be initial iterates for BNP applied to the related ordinary least-squares problem defined in Equations* (26) *through* (33). *Then subsequent iterates satisfy* $y^{(k)} = y_p + Nx^{(k)}$.

Proof. Refer to Algorithm 1. Remembering that a tilde ($\tilde{}$) represents a quantity associated with the ordinary least-squares problem, we prove by induction that $y^{(k)} = y_p + Nx^{(k)}$, $r_2^{(k)} = \tilde{r}_2^{(k)}$, $r_1^{(k)} = \tilde{r}_1^{(k)}$, $\nu_k = \tilde{\nu}_k$, and $s_k = \tilde{s}_k$ for all k.

First consider $k = 0$. We know $y^{(0)} = y_p + Nx^{(0)}$ by the definition of $x^{(0)}$. Since $\tilde{A}_1 = I$ and $\tilde{b}_1 = 0$, we have $\tilde{r}_1 = -x^{(0)}$, which is $-\overline{K}^T(A_1 y^{(0)} - b_1)$. But $\overline{K}^T A_1 = G_1$ and $\overline{K}^T b_1 = c_1$, so $\tilde{r}_1^{(0)} = c_1 - G_1 y^{(0)}$, which is $r_1^{(0)}$ as required. Additionally, simple substitution tells us that $\tilde{r}_2^{(0)}$ is $c_2 - A_2(y_p + Nx^{(0)})$. This is $c_2 - A_2 y^{(0)}$, which is $r_2^{(0)}$ as required. Finally, note that the initial defects and initial direction vectors satisfy $\nu_0 = \tilde{\nu}_0$ and $s_0 = \tilde{s}_0$ trivially.

Now assume $y^{(k)} = y_p + Nx^{(k)}$, $r_2^{(k)} = \tilde{r}_2^{(k)}$, $r_1^{(k)} = \tilde{r}_1^{(k)}$, $\nu_k = \tilde{\nu}_k$, and $s_k = \tilde{s}_k$ are all true for a fixed k, and consider $k + 1$. When Algorithm 1 is applied to the ordinary least-squares problem, note that $\tilde{Y} = Y$. This, combined with the inductive assumptions, gives us $\gamma_k = \tilde{\gamma}_k$ immediately, from which we get $r_2^{(k+1)} = \tilde{r}_2^{(k+1)}$, $r_1^{(k+1)} = \tilde{r}_1^{(k+1)}$, and $\nu_{k+1} = \tilde{\nu}_{k+1}$. It then follows that $\beta_{k+1} = \tilde{\beta}_{k+1}$, from which we obtain $s_{k+1} = \tilde{s}_{k+1}$ as required.

Finally, we know $y^{(k+1)} = y^{(k)} + \gamma_k A_1^{-1}\overline{K}s_k$. Recalling that $\tilde{A}_1 = I$, we have $A_1^{-1}\overline{K} = \tilde{A}_1^{-1}N$. Moreover, $s_k = \tilde{s}_k$ and $y^{(k)} = y_p + Nx^{(k)}$ by hypothesis. So $y^{(k+1)} = y_p + N(x^{(k)} + \gamma_k \tilde{A}_1^{-1}\tilde{s}_k)$, which is $y_p + Nx^{(k+1)}$ as required. ∎

LEMMA 3. *Let* $y^{(0)}$, $r_2^{(0)} = c_2 - A_2 y^{(0)}$, $r_1^{(0)} = c_1 - G_1 y^{(0)}$, *and* $\lambda^{(0)}$ *be initial iterates for* 3-*AOR applied to problem LSE. Define* $x^{(0)}$ *as in Equation*

(34), *and let* $x^{(0)}$, $\tilde{r}_2^{(0)} = \tilde{b}_2 - \tilde{A}_2 x^{(0)}$, *and* $\tilde{r}_1^{(0)} = \tilde{b}_1 - \tilde{A}_1 x^{(0)}$ *be initial iterates for* 3-AOR (*with the same choice of parameters*) *applied to the related ordinary least-squares problem defined by Equations* (26) *through* (33). *Then subsequent iterates satisfy* $y^{(k)} = y_p + Nx^{(k)}$.

Proof. Refer to Algorithms 2 and 3. We will establish by induction that $y^{(k)} = y_p + Nx^{(k)}$, $r_2^{(k)} = \tilde{r}_2^{(k)}$, and $r_1^{(k)} = \tilde{r}_1^{(k)}$. The case $k = 0$ is in the proof of Lemma 2.

Now assume $y^{(k)} = y_p + Nx^{(k)}$, $r_2^{(k)} = \tilde{r}_2^{(k)}$, and $r_1^{(k)} = \tilde{r}_1^{(k)}$ are true for a fixed k, and consider $k + 1$. Since $\tilde{A}_1 = I$ and $\tilde{b} = 0$, we know that $x^{(k+\frac{1}{2})} = -\tilde{r}_1^{(k)}$ for all k. By the inductive assumptions, this means that $x^{(k+\frac{1}{2})} = -r_1^{(k)}$. Thus the defining equation $y^{(k+\frac{1}{2})} = A_1^{-1}(b_1 - \bar{K}r_1^{(k)})$ becomes $y^{(k+\frac{1}{2})} = y_p + Nx^{(k+\frac{1}{2})}$. We also know that $y^{(k)} = y_p + Nx^{(k)}$ by inductive assumption. Thus, the equation $y^{(k+1)} = (1-\omega)y^{(k)} + \omega y^{(k+\frac{1}{2})}$ can be written

$$y^{(k+1)} = y_p + N\{(1-\omega)x^{(k)} + \omega x^{(k+\frac{1}{2})}\},$$

which is $y_p + Nx^{(k+1)}$ as required.

We obtain $r_2^{(k+1)} = \tilde{r}_2^{(k+1)}$ by a straightforward substitution. Observing that $r_1^{(k)} = \bar{K}^T z_3^{(k)}$, a simple substitution produces $r_2^{(k+1)} = \tilde{r}_2^{(k+1)}$ as well. ■

LEMMA 4. *Let* $y^{(0)}$, $r_2^{(0)} = c_2 - A_2 y^{(0)}$, *and* $r_1^{(0)} = c_1 - G_1 y^{(0)}$ *be initial iterates for either BNP or* 3-AOR *applied to problem LSE* (*the choice of* $\lambda^{(0)}$ *in* 3-AOR *is immaterial*). *Define* $x^{(0)}$ *as in Equation* (34), *and let* $x^{(0)}$, $\tilde{r}_2^{(0)} = \tilde{b}_2 - \tilde{A}_2 x^{(0)}$, *and* $\tilde{r}_1^{(0)} = \tilde{b}_1 - \tilde{A}_1 x^{(0)}$ *be initial iterates for the same algorithm applied to the related ordinary least-squares problem defined by Equations* (26) *through* (33). *Then the residuals* $c - Gy^{(k)}$ *and* $\tilde{b} - \tilde{A}x^{(k)}$ *are equal.*

Proof. By Lemmas 2 and 3, we know that $y^{(k)} = y_p + Nx^{(k)}$. This means that $c - Gy^{(k)} = (c - Gy_p) - GNx^{(k)}$. But $GN = \tilde{A}$ and $c - Gy_p = \tilde{b}$ by (26), so the result follows. ■

THEOREM 2. *Let* $y^{(0)}$, $r_2^{(0)} = c_2 - A_2 y^{(0)}$, *and* $r_1^{(0)} = c_1 - G_1 y^{(0)}$ *be initial iterates for BNP applied to problem LSE. Let the same quantities with an arbitrary* $\lambda^{(0)}$ *serve as initial iterates for* 3-AOR. *Let* $y_{\text{BNP}}^{(k)}$ *and* $y_{\text{AOR}}^{(k)}$ *represent the* kth *solution iterates generated by the two algorithms respectively, and*

define $r_{\mathrm{BNP}}^{(k)} = c - Gy_{\mathrm{BNP}}^{(k)}$ *and* $r_{\mathrm{AOR}}^{(k)} = c - Gy_{\mathrm{AOR}}^{(k)}$ *to be the associated residuals. Then the iterates satisfy the inequality*

$$\left\| r_{\mathrm{BNP}}^{(k)} \right\|_2 \leqslant \left\| r_{\mathrm{AOR}}^{(k+1)} \right\|_2$$

in exact arithmetic, regardless of the AOR iteration parameters.

Proof. Apply BNP and 3-AOR to the related ordinary least-squares problem defined by Equations (26) through (33). By Lemma 4 we have

$$r_{\mathrm{BNP}}^{(k)} = \tilde{b} - \tilde{A}x_{\mathrm{BNP}}^{(k)},$$

$$r_{\mathrm{AOR}}^{(k+1)} = \tilde{b} - \tilde{A}x_{\mathrm{AOR}}^{(k+1)}.$$

The result then follows from Theorem 1. ■

While the theorem applies to calculations performed in exact arithmetic, it suggests that BNP should outperform 3-AOR on problem LSE. Our numerical experiments suggest that this is in fact the case.

6. NUMERICAL EXPERIMENTS

In this section, we compare the performance of BNP, 2-cyclic SOR, and 3-AOR on the three structural engineering problems shown in Figure 1. Problem WRENCH [Figure 1(a)] is one of a suite of six test problems

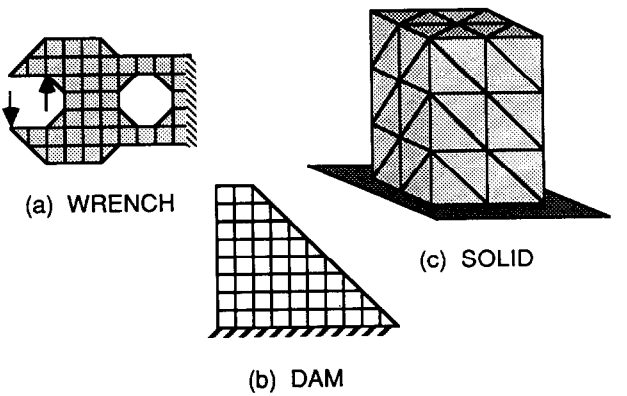

FIG. 1. Structural engineering test problems.

developed by M. Lawo [3]. It consists of 48 planar elements, and leads to an LSE problem with 112 constraints and 216 unknown internal forces; the diagonal blocks in the element flexibility matrix F are 5×5 for the square elements and 3×3 for the triangular elements. The applied force is indicated by the arrows in the figure. Problem DAM [Figure 1(b)], modeled by the author using techniques described in Przemieniecki [16], is intended to approximate a cross-section of a dam subjected to the force of a body of water against its left wall. This version of DAM consists of 52 planar elements, and leads to a problem with 104 constraints and 244 unknowns. Problem SOLID [Figure 1(c)], also modeled by the author using techniques in [16], approximates a building subjected to the force of a steady wind approaching one of its vertical edges. This version consists of 60 solid tetrahedral elements, and produces a problem with 81 constraints and 360 unknowns. Each block in the block-diagonal matrix F is 6×6. See [10] for experiments on larger versions of these problems, including structures with simulated damage.

All experiments were run on a single processor of a two-processor Alliant FX/40 (see [9] and [10] for results using parallel versions of the algorithms on substructured problems). We use sparse data structures and double-precision arithmetic in all calculations. To measure error, we obtained the "true" solution to each problem by solving the original Kuhn-Tucker equations using LINPACK [4]. We then adjusted the stop tolerances for each experiment so that the infinity norm of the error was roughly 1×10^{-4}. The execution times (in seconds), obtained using the Alliant `etime` intrinsic, include all operations except input/output. In all problems, however, only the iteration times were significant: preprocessing, including factoring E and forming A_1, typically required only 1–2% of the cpu time.

The results for SOR and AOR are for the approximate optimal values of the iteration parameters (obtained experimentally). We report that AOR is highly sensitive to the choice of the parameters: very small deviations from the optimal values result in either divergence or a drastic reduction in the rate of convergence. In all three test problems, the region of convergence in the ω-β plane appears to be a tiny, narrow crescent-shaped region. The choice of parameter in 2-cyclic SOR is consistent with the theory (see [1] and [12]): convergence occurs for all ω below a sufficiently small critical value, with the optimal choice occurring very close to that critical value. It is clear from Table 1 that these test problems do not satisfy the conditions in Papadopoulou, Saridakis, and Papatheodorou [13], since 2-SOR outperforms 3-AOR by a wide margin.

The results on all three test problems suggest the theoretical results are highly conservative: BNP outperforms both optimal 2-cyclic SOR and optimal 3-AOR by a wide margin. If anything, the experiments may understate the advantage of BNP, since production codes would normally need to

TABLE 1
SUMMARY OF NUMERICAL RESULTS

	WRENCH	DAM	SOLID
m_1	112	104	81
n	216	244	360
Iterations:			
BNP	33	51	41
2-SOR	457	818	208
3-AOR	2439	5991	609
Time (sec):			
BNP	0.48	0.899	1.08
2-SOR	5.79	12.4	4.81
3-AOR	19.1	91.7	14.0

determine the SOR and AOR parameters adaptively. While these tests are hardly exhaustive, we believe the conclusion is clear: both theory and practice indicate that the order-reducing conjugate-gradient algorithm proposed by Barlow, Nichols, and Plemmons offers a competitive approach to solving problems of this type.

REFERENCES

1 J. Barlow, N. Nichols, and R. Plemmons, Iterative methods for equality constrained least squares problems, *SIAM J. Sci. Statist. Comput.* 9:892–906 (1988).
2 J. Barlow, N. Nichols, and R. Plemmons, Iterative Methods for Equality Constrained Least Squares Problems, Internal Report CS-87-04, Dept. of Computer Science, Pennsylvania State Univ., Jan. 1987.
3 M. Berry, M. Heath, I. Kaneko, M. Lawo, R. Plemmons, and R. Ward, An algorithm to compute a sparse basis of the null space, *Numer. Math.* 47:483–504 (1985).
4 J. Dongarra, J. Bunch, C. Moler, and G. Stewart, *LINPACK Users' Guide*, SIAM, Philadelphia, 1979.
5 M. Eiermann, W. Niethammer, and A. Ruttan, Optimal successive overrelaxation iterative methods for p-cyclic matrices, *Numer. Math.*, to appear.
6 R. Freund, A note on two block-SOR methods for sparse least squares problems, *Linear Algebra Appl.* 88–89:211–221 (1987).
7 A. Hadjidimos, Accelerated overrelaxation method, *Math. Comp.* 32:149–157 (1978).
8 M. Heath, R. Plemmons, and R. Ward, Sparse orthogonal schemes for structural optimization using the force method, *SIAM J. Sci. Statist. Comput.* 5(3):514–532 (1988).

9 D. James, Implicit nullspace iterative methods for constrained least squares problems, *SIAM J. Matrix Analysis and Appl.*, submitted for publication.
10 D. James, Conjugate Gradient Methods for Constrained Least Squares Problems, Ph.D. Dissertation, Dept. of Mathematics, North Carolina State Univ., Raleigh, 1990.
11 D. James and R. Plemmons, An iterative substructuring algorithm for equilibrium equations, *Numer. Math.*, 57:625–633 (1990).
12 T. Markham, M. Neumann, and R. Plemmons, Convergence of a direct-iterative method for large scale least squares problems, *Linear Algebra Appl.* 69:155–167 (1985).
13 E. Papadopoulou, Y. Saridakis and T. Papatheodorou, Block AOR iterative schemes for large-scale least-squares problems, *SIAM J. Numer. Anal.* 26(3): 637–660 (June 1989).
14 D. Pierce, A. Hadjidimos, and R. Plemmons, Optimality relationships for p-cyclic SOR, *Numer. Math.*, 56:635–643 (1990).
15 R. Plemmons, A parallel block iterative scheme applied to computations in structural analysis, *SIAM J. Algebraic Discrete Methods* 7(3):337–347 (July 1986).
16 J. Przemieniecki, *Theory of Matrix Structural Analysis*, Dover, 1985.
17 G. Strang, A framework for equilibrium equations, *SIAM Rev.* 30(2):283–297 (June 1988).
18 G. Strang, *Introduction to Applied Mathematics*, Wellesley Cambridge Press, Wellesley, Mass, 1986.
19 R. Varga, p-cyclic matrices: A generalization of the Young-Frankel successive overrelaxation scheme, *Pacific J. Math.* 9:617–623 (1959).
20 R. Varga, *Matrix Iterative Analysis*, Prentice-Hall, Englewood Cliffs, N.J., 1963.
21 D. Young, Iterative methods for solving partial differential equations of elliptic type, *Trans. Amer. Math. Soc.* 76:92–111 (1954).

Received 3 January 1990; final manuscript accepted 16 Febraury 1990